IMPRS - BBL

# Numerical methods

Lecture 5

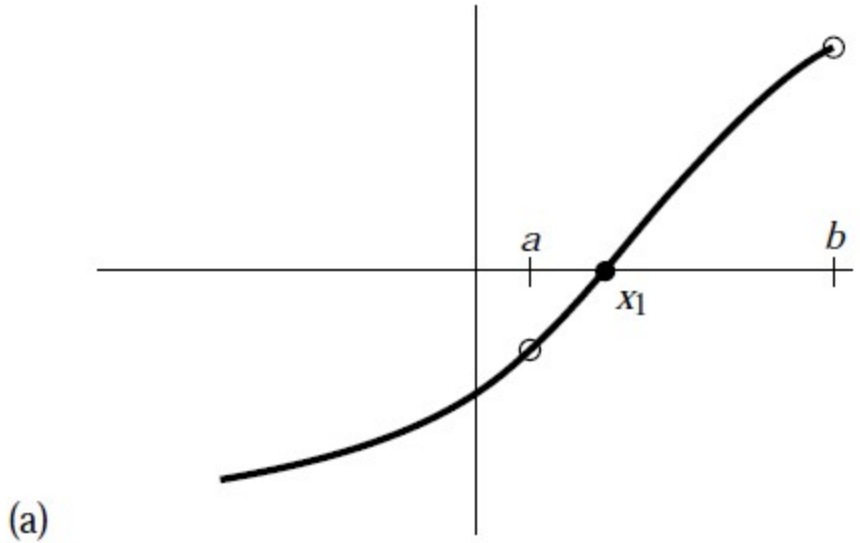
Michael Marks

# Topics:

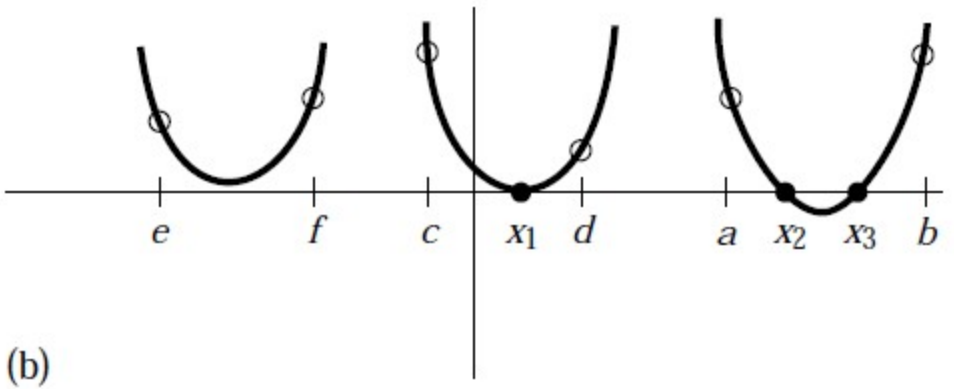
Day 1: Linear algebraic equations
Day 2: Inter- and Extrapolation
Day 3: Integration
Day 4: Random numbers and distribution functions
Day 5: Root finding, Minimization and Maximization
Day 6: Differentiation

# Root finding

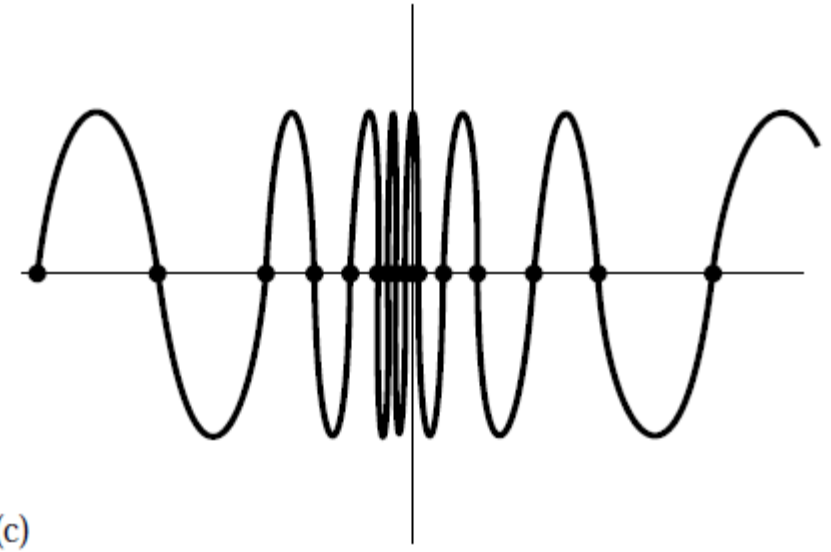
situations



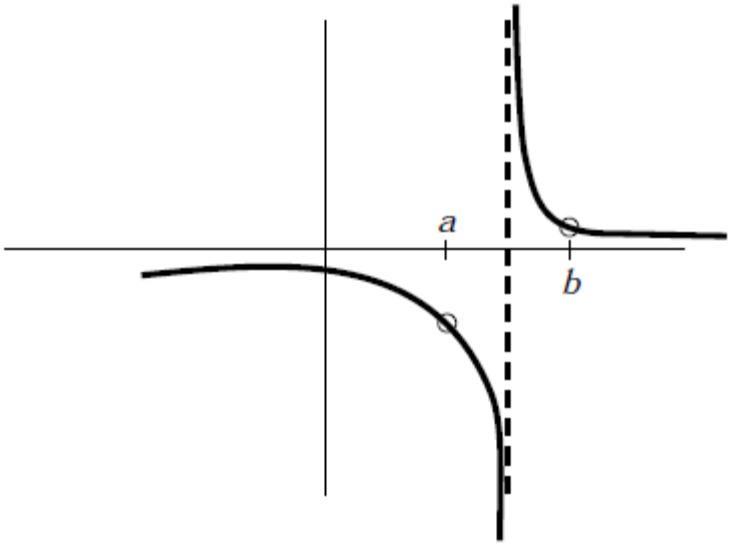

(a)




(b)



(c)




(d)

Numerical recipes

# Root finding

Bracketing

```
#include <math.h>
#define FACTOR 1.6
#define NTRY 50

int zbrac(float (*func)(float), float *x1, float *x2)
```

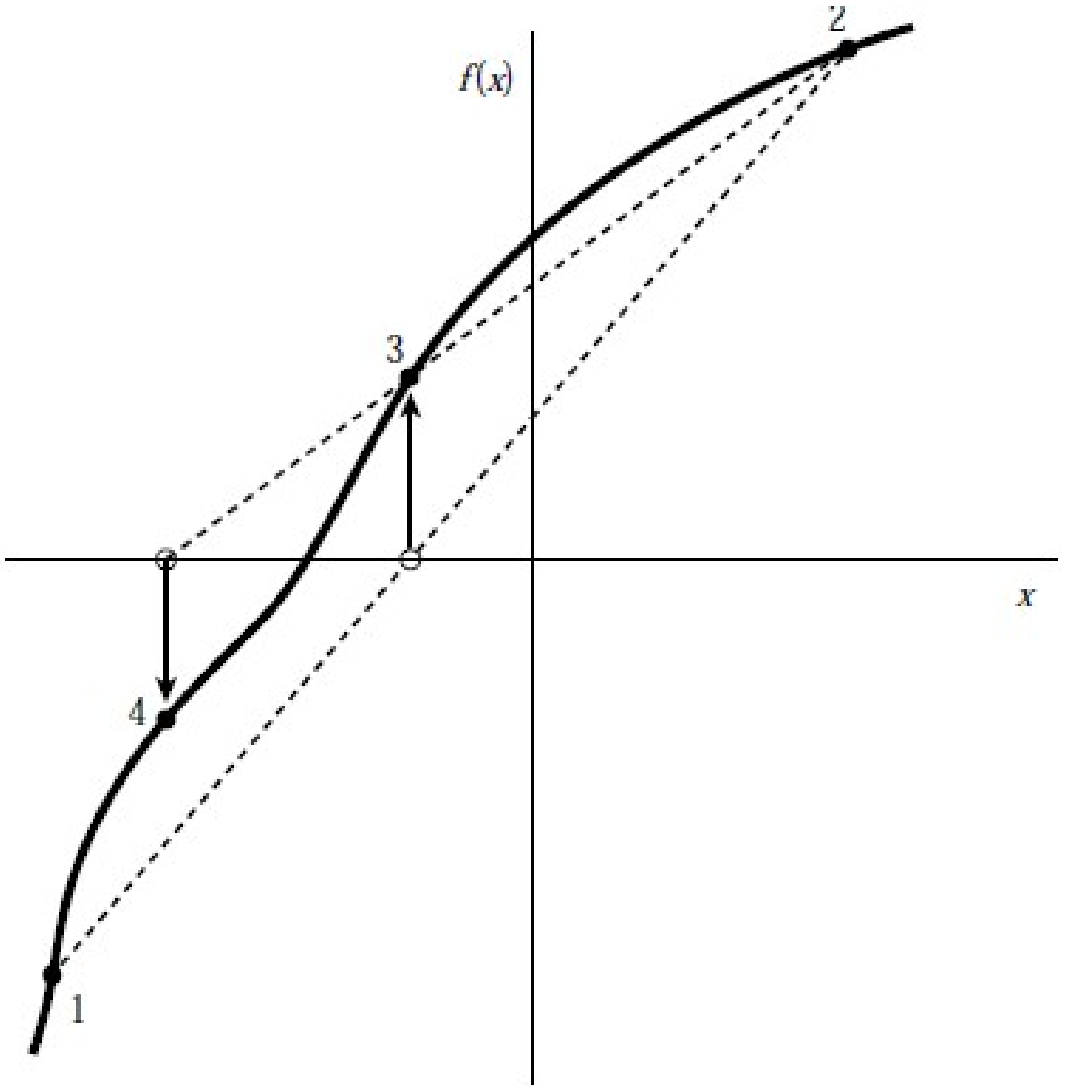
Given a function `func` and an initial guessed range `x1` to `x2`, the routine expands the range geometrically until a root is bracketed by the returned values `x1` and `x2` (in which case `zbrac` returns 1) or until the range becomes unacceptably large (in which case `zbrac` returns 0).

```
{
    void nrerror(char error_text[]);
    int j;
    float f1,f2;

    if (*x1 == *x2) nrerror("Bad initial range in zbrac");
    f1=(*func)(*x1);
    f2=(*func)(*x2);
    for (j=1;j<=NTRY;j++) {
        if (f1*f2 < 0.0) return 1;
        if (fabs(f1) < fabs(f2))
            f1=(*func)(*x1 += FACTOR*(*x1-*x2));
        else
            f2=(*func)(*x2 += FACTOR*(*x2-*x1));
    }
    return 0;
}
```

Numerical recipes

# Root finding

Secant method




Numerical recipes

# Root finding

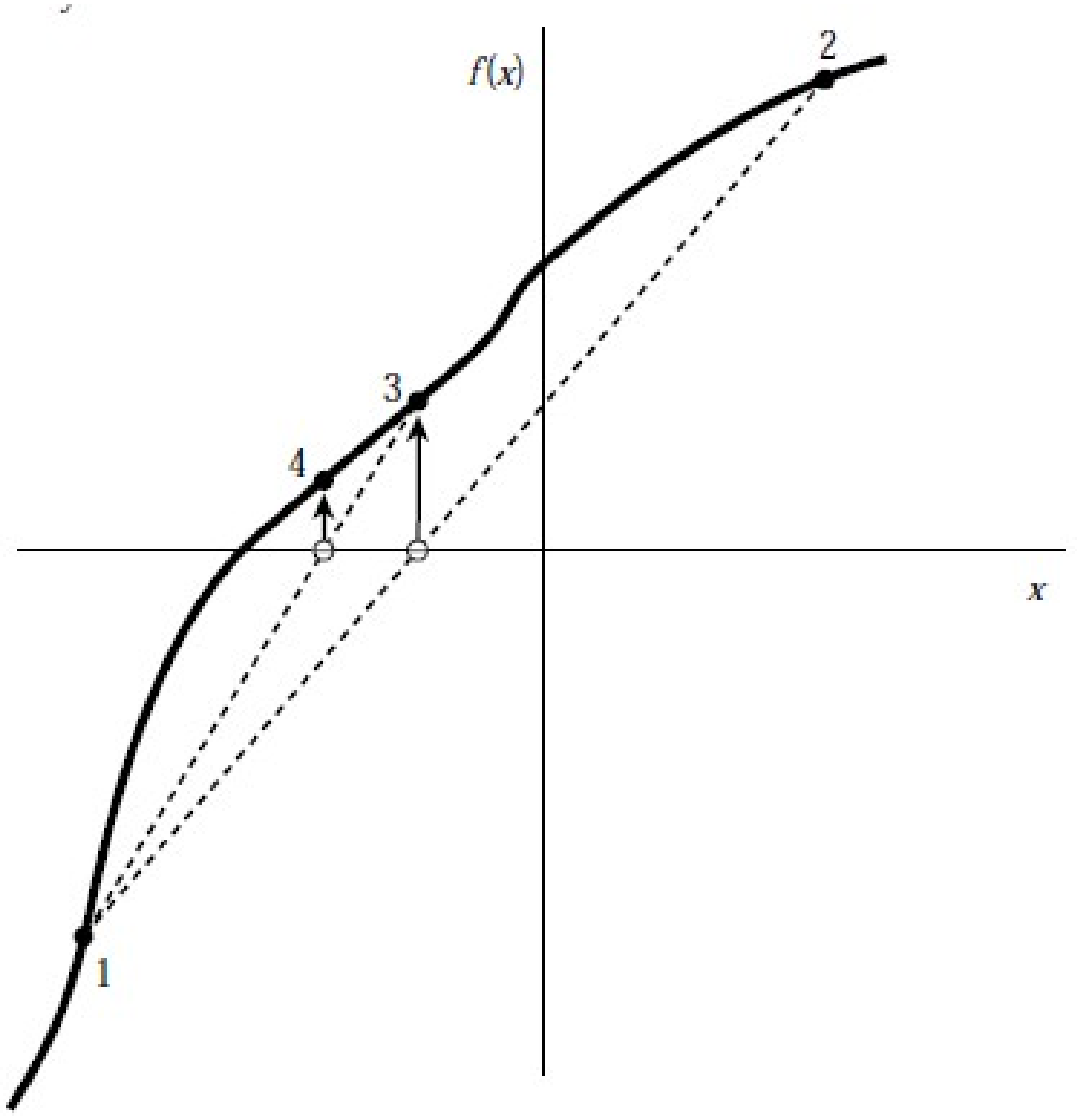
## Secant method

```
#include <math.h>
#define MAXIT 30                        Maximum allowed number of iterations.

float rtsec(float (*func)(float), float x1, float x2, float xacc)
Using the secant method, find the root of a function func thought to lie between x1 and x2.
The root, returned as rtsec, is refined until its accuracy is ±xacc.
{
    void nrerror(char error_text[]);
    int j;
    float fl,f,dx,swap,xl,rts;

    fl=(*func)(x1);
    f=(*func)(x2);
    if (fabs(fl) < fabs(f)) {           Pick the bound with the smaller function value as
        rts=x1;                             the most recent guess.
        xl=x2;
        swap=fl;
        fl=f;
        f=swap;
    } else {
        xl=x1;
        rts=x2;
    }
    for (j=1;j<=MAXIT;j++) {            Secant loop.
        dx=(xl-rts)*f/(f-fl);           Increment with respect to latest value.
        xl=rts;
        fl=f;
        rts += dx;
        f=(*func)(rts);
        if (fabs(dx) < xacc || f == 0.0) return rts;     Convergence.
    }
    nrerror("Maximum number of iterations exceeded in rtsec");
    return 0.0;                         Never get here.
}
```

Numerical recipes

# Root finding

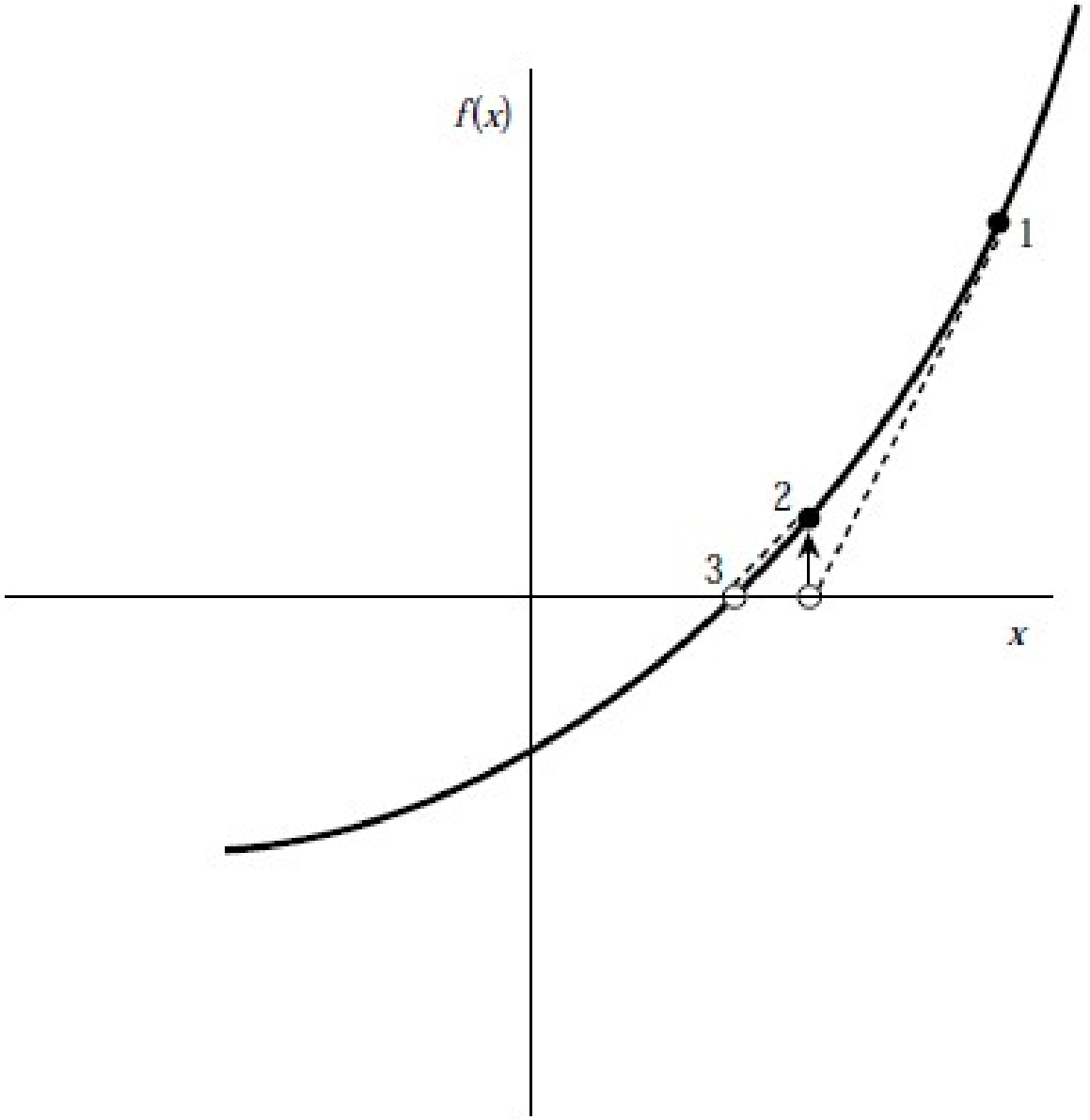
False position method




Numerical recipes

# Root finding

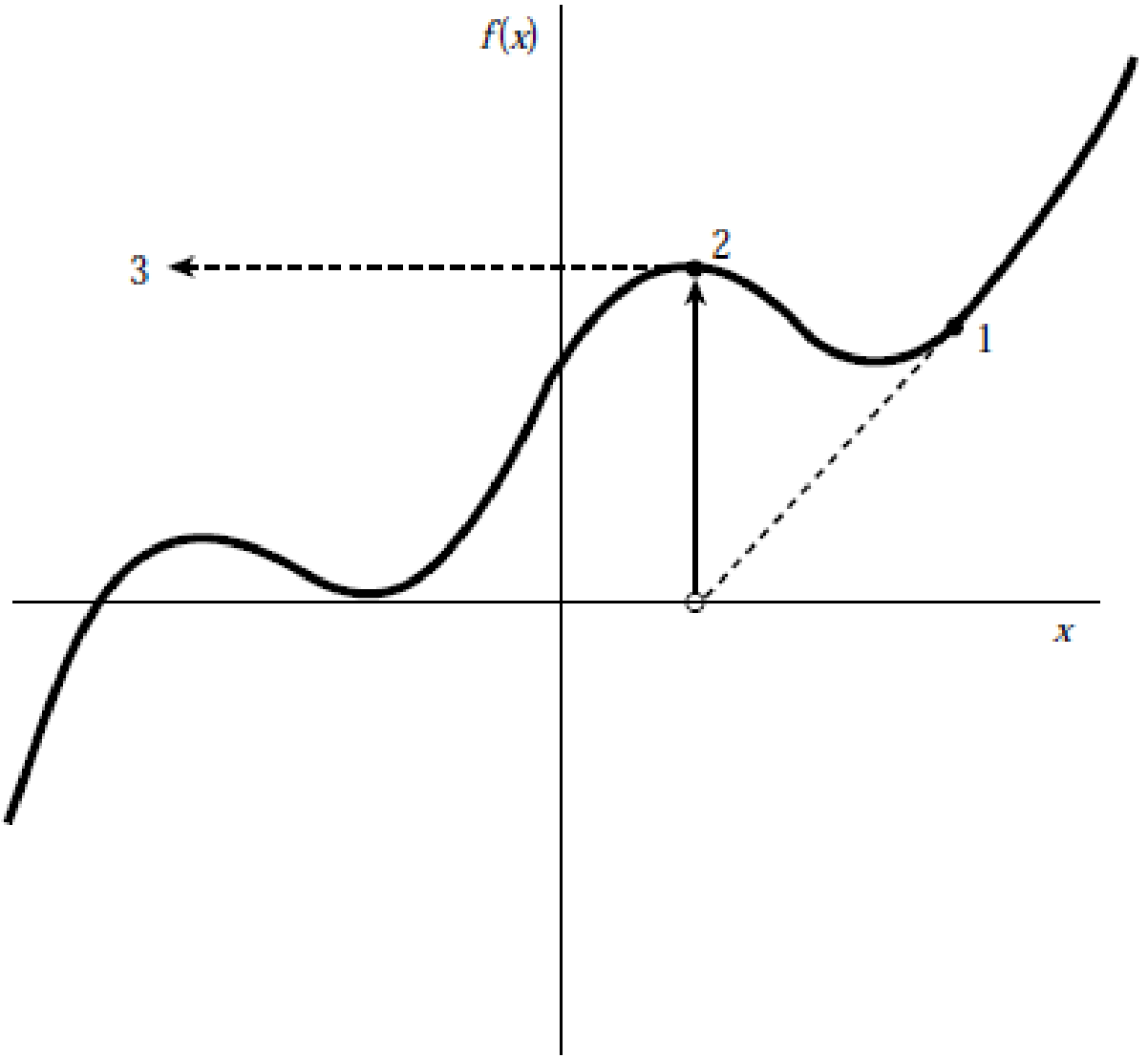
Newton Raphson Method




Numerical recipes

# Root finding

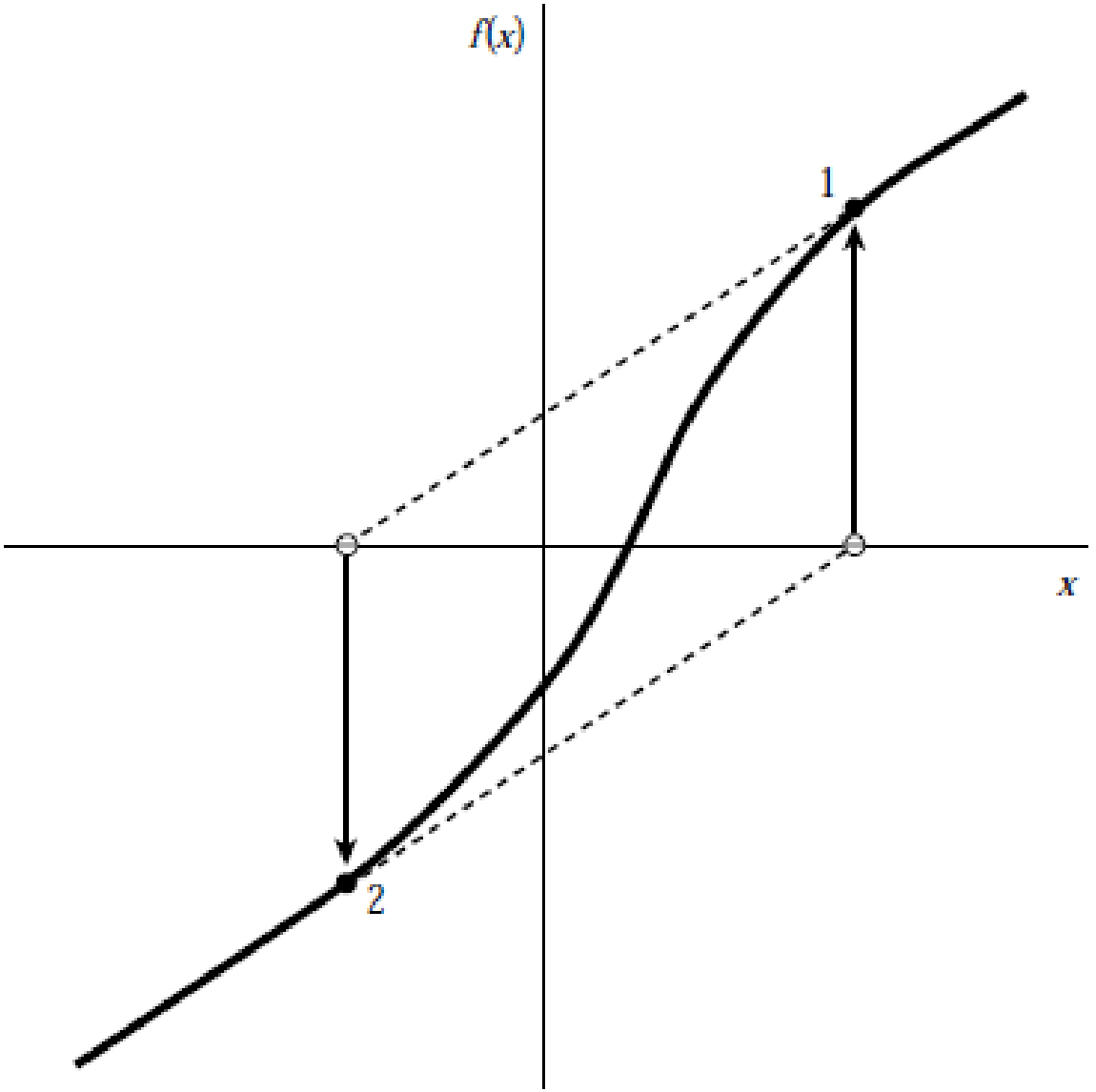
Newton Raphson Method - Problem




Numerical recipes

# Root finding

Newton Raphson Method - Problem




Numerical recipes

# Root finding

## Newton Raphson Method

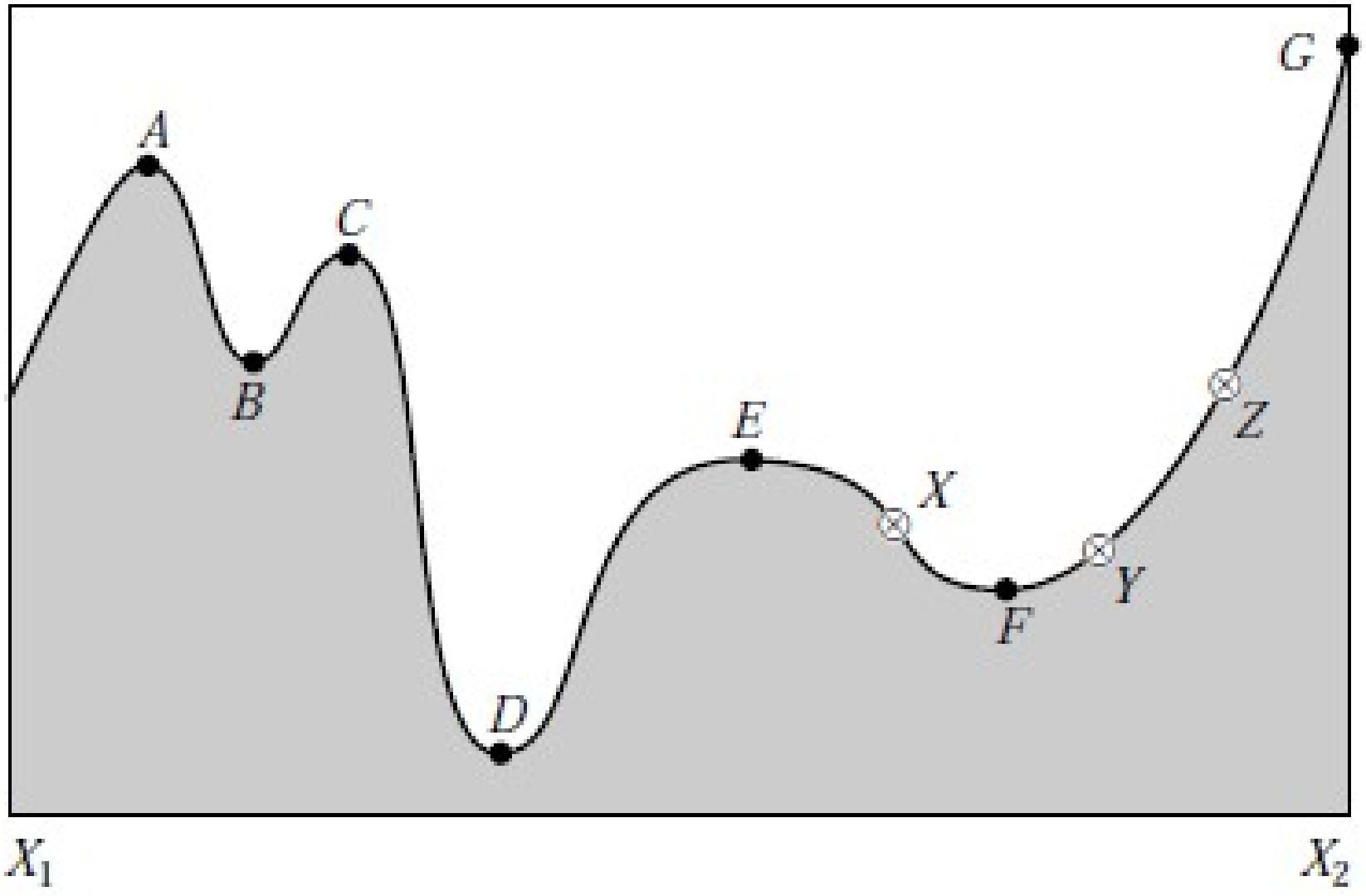
```
#include <math.h>
#define JMAX 20                                   Set to maximum number of iterations.

float rtnewt(void (*funcd)(float, float *, float *), float x1, float x2,
    float xacc)
```

Using the Newton-Raphson method, find the root of a function known to lie in the interval [x1,x2]. The root `rtnewt` will be refined until its accuracy is known within ±`xacc`. `funcd` is a user-supplied routine that returns both the function value and the first derivative of the function at the point `x`.

```
{
    void nrerror(char error_text[]);
    int j;
    float df,dx,f,rtn;

    rtn=0.5*(x1+x2);                              Initial guess.
    for (j=1;j<=JMAX;j++) {
        (*funcd)(rtn,&f,&df);
        dx=f/df;
        rtn -= dx;
        if ((x1-rtn)*(rtn-x2) < 0.0)
            nrerror("Jumped out of brackets in rtnewt");

        if (fabs(dx) < xacc) return rtn;          Convergence.
    }
    nrerror("Maximum number of iterations exceeded in rtnewt");
    return 0.0;                                   Never get here.
}
```

Numerical recipes

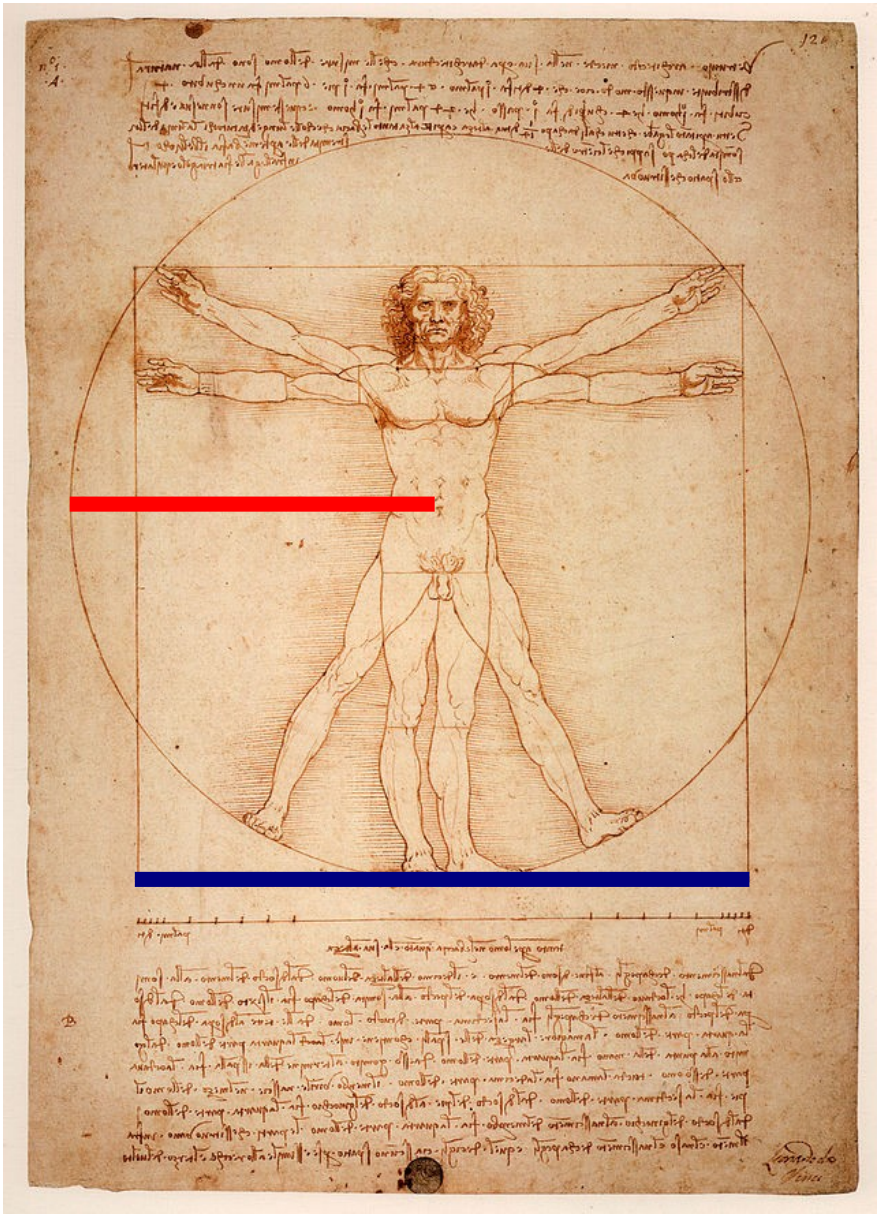
# Maximiziation / Minimization



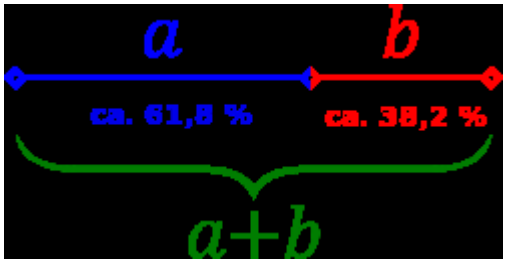

Numerical recipes

# Maximiziation / Minimization

Golden Section Search



The Vitruvian Man,
da Vinci




Wikipedia